

Tutorial - Track 5: The Effective Use of System and Software Architecture Standards for Software Technology Readiness Assessments

Dr. Peter Hantos
The Aerospace Corporation

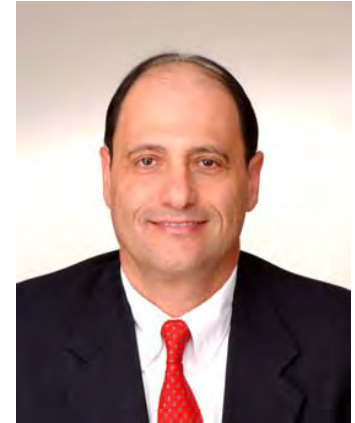
Systems & Software Technology Conference, Salt Lake City, UT
May 16, 2011

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE MAY 2011		2. REPORT TYPE		3. DATES COVERED 00-00-2011 to 00-00-2011	
4. TITLE AND SUBTITLE The Effective Use of System and Software Architecture Standards for Software Technology Readiness Assessments				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) The Aerospace Corporation,P.O. Box 92957-M1/112,Los Angeles,CA,90009-2957				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited					
13. SUPPLEMENTARY NOTES Presented at the 23rd Systems and Software Technology Conference (SSTC), 16-19 May 2011, Salt Lake City, UT. Sponsored in part by the USAF. U.S. Government or Federal Rights License					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report (SAR)	18. NUMBER OF PAGES 77	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

About Your Instructor

Dr. Peter Hantos

Senior Engineering Specialist
The Aerospace Corporation
P.O. Box 92957-M1/112
Los Angeles, CA 90009-2957
Phone: (310) 336-1802
E-Mail: peter.hantos@aero.org



Degrees Received:

Doctorate in Automation, Technical University, Budapest, Hungary
M.S. in Electrical Engineering, Technical University, Budapest, Hungary

Industry Experience:

Over 35 years of combined experience as professor, researcher, software engineer and manager, with accomplishments in software engineering, manufacturing automation, office automation and signal processing. As a senior software engineer at Xerox he was member of the original development team that was chartered to bring into the product development domain the world-known inventions from the Xerox Palo Alto Research Center (PARC), primarily the use of the desktop, icons, mouse-control and network paradigms. Successfully directed engineering and quality process development on all levels of the enterprise. As principal scientist of the Xerox Corporate Software Engineering Center developed and implemented two corporate-wide processes for software-intensive product development and the assessment of software technology readiness for productization. In the same capacity he also implemented a corporate-wide software risk assessment program.

Government and Defense Program Support Experience:

USAF, NASA, FBI, CIA, and other, miscellaneous government and commercial customers of The Aerospace Corporation

Teaching Experience:

Tutorials: GSAW 2005, 2006, 2007, 2008, 2009, 2010, 2011; Euro-SEPG 2005; INCOSE 2005; NDIA 2008, 2009, 2010; and SSTC 2003, 2005, and 2010. "Space System Software Project Management", "Space System Software Acquisition Management", "Space System Software Product Development", "Space System Development, Integration, & Test", "Introduction to Program Office Data and Controls", and "Program Measurement Workshop" courses for The Aerospace Institute; Electrical Engineering and Computer Science courses at the Technical University, Budapest, Hungary and at the University of California, Santa Barbara. Dr. Hantos has also been authorized by the SEI to teach the "Introduction to CMMI®-DEV V1.2" SEI Course.

Acknowledgements

- This work would not have been possible without the help of the following people of The Aerospace Corporation
 - *Asya Campbell*
 - *John C. Cantrell*
 - *Suellen Eslinger*
 - *B. Zane Faught*
 - *Dr. Leslie J. Holloway*
- Funding Source
 - The Aerospace Corporation's Independent Research & Development Program
- A special thanks goes to the members of the Air Force Smart Operations for the 21st Century/Develop and Sustain Warfighting Systems/Technology Development Team
 - *Dr. Thomas Christian (AFMC/ASC)*
 - *Suzanne Garcia (SEI)*
 - *William Nolte (AFMC/AFRL)*
 - *Dr. Paul Phister (AFMC/AFRL)*
 - *Dr. Kyle Yang (MIT/Lincoln Lab)*

Challenges, Challenges, Challenges ...

Technology

“Any sufficiently advanced technology is indistinguishable from magic.”

~~~ Arthur C. Clarke

Technology Readiness

When the nation's first ballistic missile rose about 6 inches above the launch pad before toppling over and exploding, <Simon> Ramo reportedly turned to an Air Force general and said: "Well, Benny, now that we know the thing can fly, all we have to do is improve its range a bit."

~~~ Peter Pae, Book Review, LA Times, 7/5/09

Standards

"Standards are always out of date. That's why we call them standards."

~~~ George F. Will on "This Week with George Stephanopoulos", 4/3/05

Outline

- Motivation
- Technology Readiness Assessments – the 64,000-foot View
- Tutorial Scope
- Risks of Software CTE Identification
- Missing TRA Definitions
- Algorithms
- Department of Defense Architecture Framework Version 2.0
- Why the Work Breakdown Structure is Inadequate for CTE Identification
- The ISO/IEC 42010:2007 Architecture Standard
- Exploring ISO/IEC 10746
- Unified Modeling Language (UML)
- Proposed UML Specification of the ISO/IEC 42010 Technology Viewpoint
- Case Study – Multimedia Conferencing System – Technology Specification
- Risks of Software TRL Determination
- Exploit Available UML Dependency Information
- Using Standards in the Acquisition Context
- Concluding Thoughts
- Acronyms
- References

Motivation (Yours...)

- Why is Technology Readiness Assessment important?
 - *“The inability to define and thus measure technology readiness facilitates decisions to incorporate immature technology in system design at Milestone B which consequently leads to technical problems during System Design and Development*.” [DAPA 2006]*
- Why should it be important for you to learn about it?
 - *For one thing, it is the Law (more on this later.) Nevertheless ...*
 - *If you are in an Acquisition Program Office (APO):*
 - You might have to provide data to an independent review panel conducting a Technology Readiness Assessment (TRA)
 - *If you are in The Aerospace Corporation:*
 - You might be invited to become a member of an independent review panel
 - *If you are a Contractor:*
 - You might want to gain insight into how your proposals are evaluated

* Note that the currently used term for the referenced DoD 5000.02 phase is Engineering and Manufacturing Development (EMD)

Technology Readiness Assessments – the 64,000-foot View



- Public Law 109-163-Jan.6, 2006, Section 801

TITLE VIII—ACQUISITION POLICY, ACQUISITION MANAGEMENT, AND RELATED MATTERS

Subtitle A—Provisions Relating to Major Defense Acquisition Programs

SEC. 801. REQUIREMENT FOR CERTIFICATION BEFORE MAJOR DEFENSE ACQUISITION PROGRAM MAY PROCEED TO MILESTONE B.

(a) CERTIFICATION REQUIREMENT.—Chapter 139 of title 10, United States Code, is amended by inserting after section 2366 the following new section:

“§ 2366a. Major defense acquisition programs: certification required before Milestone B or Key Decision Point B approval

(a) CERTIFICATION.—A major defense acquisition program may not receive Milestone B approval, or Key Decision Point B approval in the case of a space program, until the milestone decision authority certifies that—

(1) the technology in the program has been demonstrated in a relevant environment; ...”

Note that the term “**Key Decision Point**” is not in use since the cancellation of NSSAP 03-01;
the phase gates for space acquisitions are also called milestones



Technology Readiness Assessments – the 64,000-foot View (Cont.)



- November 2, 2007 Air Force Memorandum on Technology Certification
 - *Spells out that for all Critical Technology Elements (CTEs) it has to be demonstrated in a relevant environment that they are at Technology Readiness Level (TRL) 6 or greater.*
- New provisions in the Weapon Systems Acquisition Reform Act of 2009

TITLE I—ACQUISITION ORGANIZATION

SEC. 104. Assessment of technological maturity of critical technologies of major defense acquisition programs by the Director of Defense Research and Engineering.

(a) ASSESSMENT BY DIRECTOR OF DEFENSE RESEARCH AND ENGINEERING.—

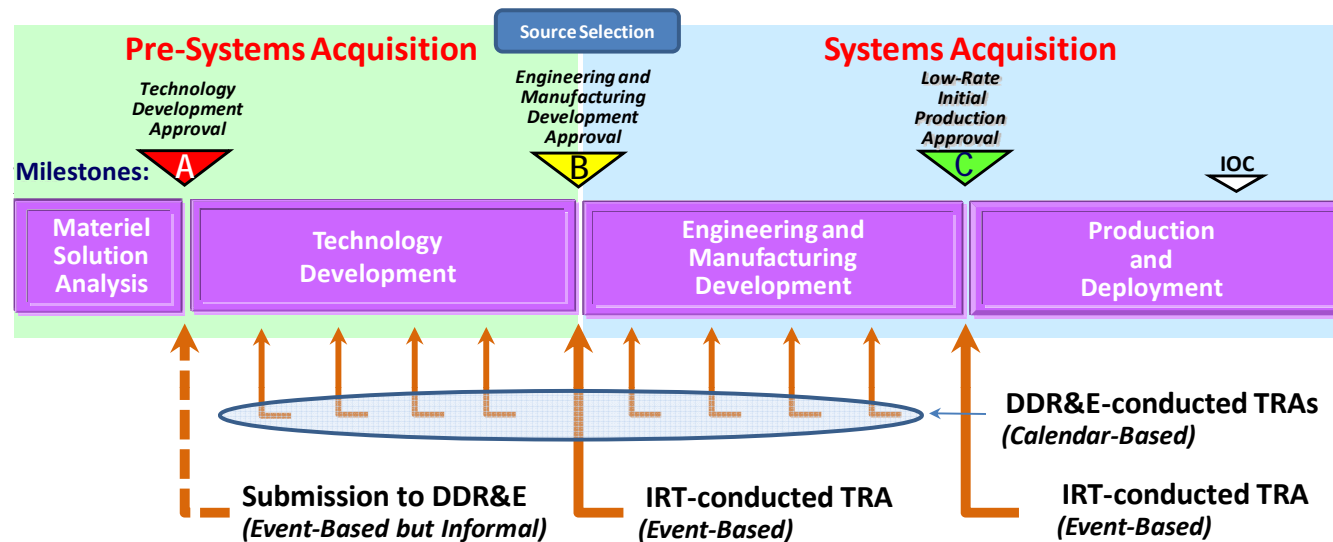
(1) IN GENERAL. — Section 139a of title 10, United States Code, is amended by adding at the end the following new subsection:

(c) (1) The Director of Defense Research and Engineering, in consultation with the Director of Developmental Test and Evaluation, shall periodically review and assess the technological maturity and integration risk of critical technologies of the major defense acquisition programs ...

(2) The Director shall submit to the Secretary of Defense and the congressional defense committees by March 1 of each year a report...

Technology Readiness Evaluation Logistics*

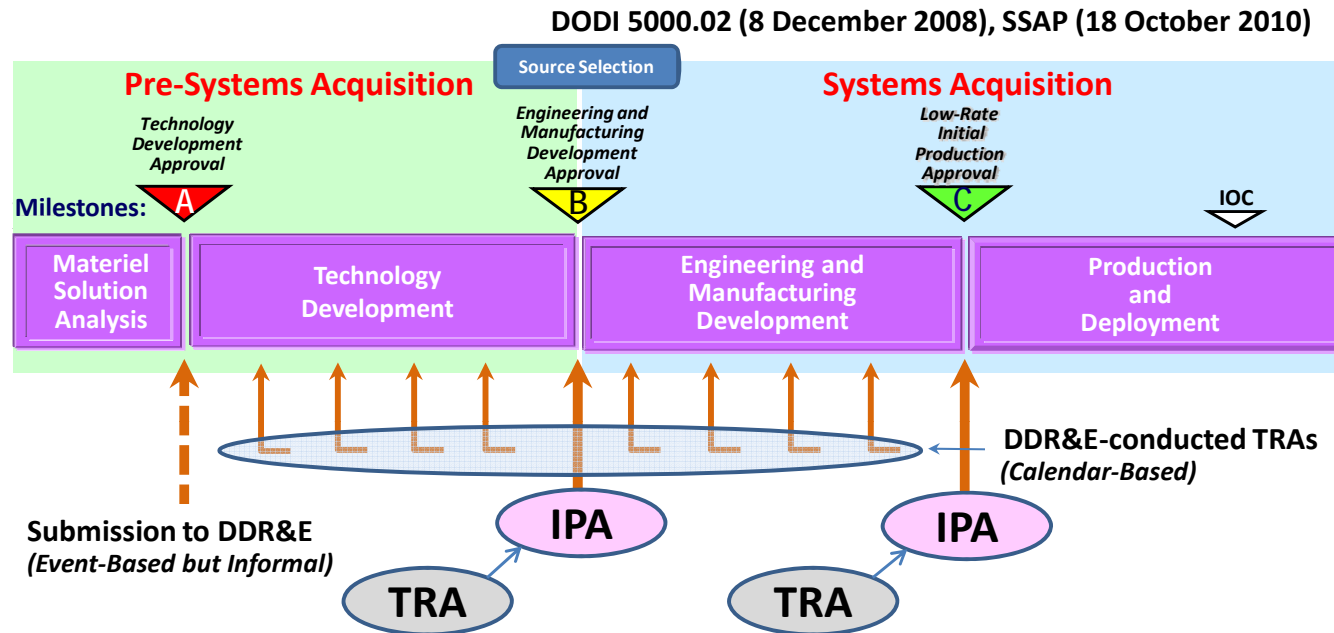
DODI 5000.02 (8 December 2008)



Steps of a formal, IRT- conducted TRA at Milestones B and C

- The Component Science & Technology Executive appoints an Independent Review Team (IRT) of appropriate Subject Matter Experts (SMEs)
- Acquisition Program Office presents its technology plans to the IRT
- IRT evaluates the plan and submits the list of selected CTEs to the Defense Acquisition Board (DAB) for approval
- IRT assesses the maturity of the approved CTEs
- IRT briefs the Milestone Decision Authority (MDA) on its findings
- MDA approves/disapproves the entry to the next acquisition phase

Technology Readiness Evaluations in Space Programs*



How is it different from the non-space DoD programs?

- The Assistant Secretary of the Air Force for Acquisition (SAF/AQ) appoints an Independent Program Assessment Team (IPAT) of appropriate Subject Matter Experts
- IPA scope is much broader than just technology – the IPAT looks at all aspects of the program, including independent cost estimates (ICEs)
- The TRA team is only a sub-team of the IPA
- The TRA team first reports its findings to the IPA (more layers...)

Basic Department of Defense (DoD) TRA Definitions

- The key document providing DoD guidance on carrying out a TRA is entitled the “Technology Readiness Assessment (TRA) Deskbook”
 - *This tutorial is based on the most recent, July 2009 edition*
- Technology* Maturity
 - *A measure or degree to which proposed technologies meet program objectives*
- Technology Readiness Assessment
 - *A TRA is a formal, systematic, metrics-based process and accompanying report that assesses the maturity of critical hardware and software technologies to be used in systems. The TRA is not intended to predict future performance of the evaluated technologies, nor does it assess the quality of the system architecture, design, or integration plan*
- TRA is different from “Conventional” Risk Management
 - *The result of a TRA is a single number on a 1-9, ordinal scale, called Technology Readiness Level (TRL).*
 - *TRLs do not intend to reflect either the likelihood of attaining required maturity or the impact of not achieving the required maturity*
- The TRA complements – but does not in any way preclude – the Program Manager’s responsibility to pursue reduction of all risks

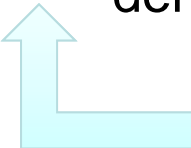
* Note that the definition of “technology” is missing

Critical Technology Elements

- Context for Technology Readiness Assessments
 - *For practical purposes not all planned technologies are assessed in a TRA*
 - The selected technologies that need to be subjected to a TRA will be called Critical Technology Elements (CTEs)
 - *However, the analysis of candidate technologies is supposed to begin even before a Materiel Development Decision takes place for the acquisition*
- A technology element is critical if
 - *The system being acquired depends on this technology element to meet operational requirements within acceptable cost and schedule limits, **and***
 - *The technology element or its application is*
 - either new or novel, **or**
 - in an area that poses major technological risk during detailed design or demonstration
- Candidate CTEs vs. CTEs
 - *Until it is approved by the Milestone Decision Authority (MDA,) all CTEs are considered only as Candidate CTEs*
- DoD guidance on sources for identifying candidate CTEs
 - *Use the DoD Architecture Framework (DoDAF) views*
 - *Use the Work Breakdown Structure (WBS)*
- Alternative USAF Recommendation [USAF 2009]:
 - *Use the IEEE Architecture Description Standard [IEEE 2000]*

Questions to be Asked to Classify Technology Elements*

- 1) Does the technology have a significant impact on an operational requirement, cost, or schedule?
- 2) Does this technology pose a major development or demonstration risk?
- 3) Is the technology new or novel?
- 4) Has the technology been modified from prior successful use?
- 5) Has the software technology been repackaged such that a new relevant environment is applicable?
- 6) Is the technology expected to operate in an environment and/or achieve a performance beyond its original design intention or demonstrated capability?



A technology element is critical if the answer to the first question and to any of the remaining questions is “yes”

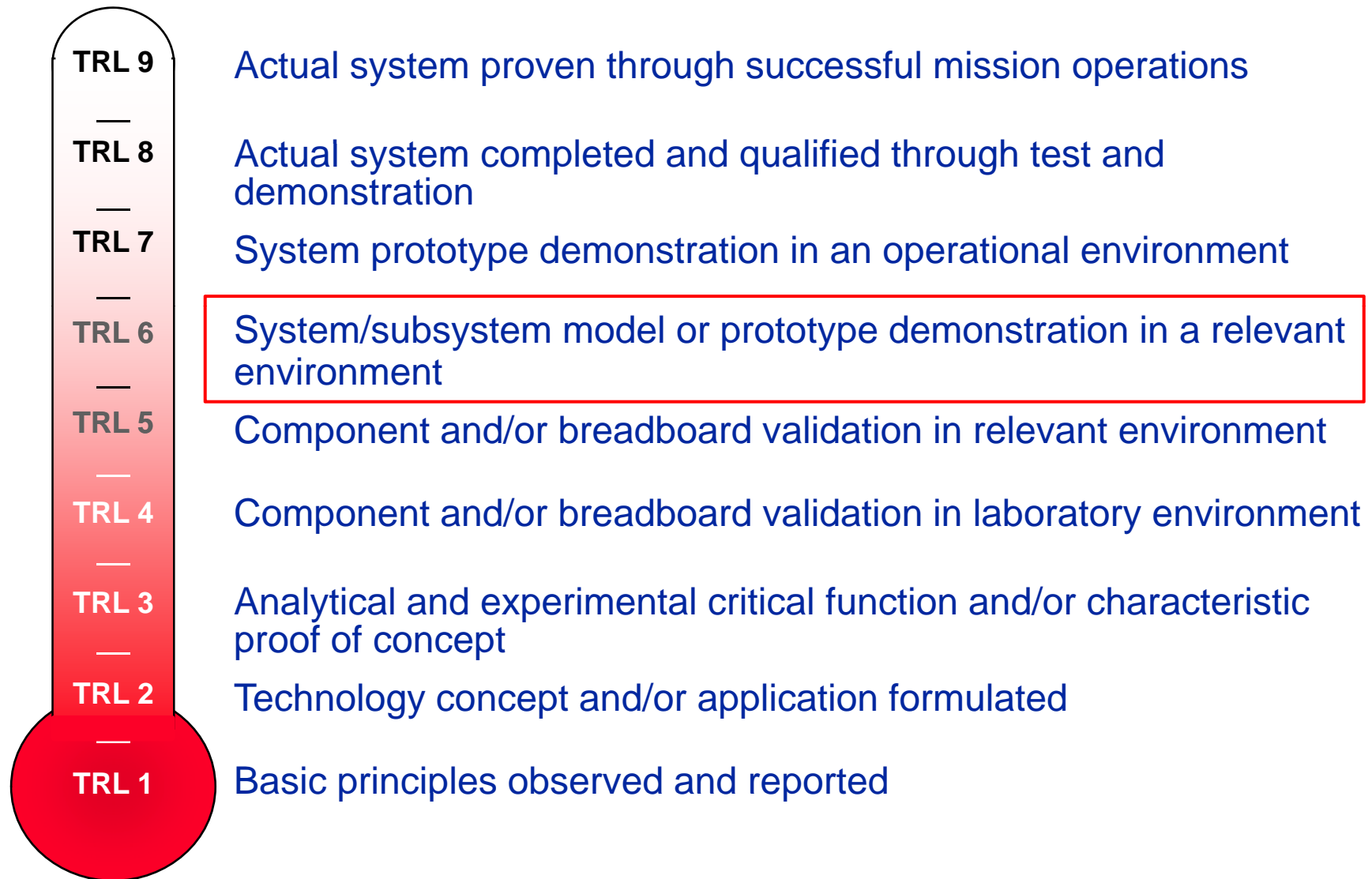
* Source: [DoD 2009], pp B-8; More details in [Hantos 2010]

Relevant Environment

- Relevant Environment Definition
 - *Relevant Environment is a validation environment that simulates key aspects of the Operational Environment*
 - *The purpose of using a relevant environment is to demonstrate sufficient confidence in the CTE; i.e., that skillful application of this technology will fully support the required threshold functionality.*
- Relevant Environment for Space*
 - *A satellite from launch to standard operation in space is exposed to drastically changing environmental conditions and a relevant environment test design must encompass all such stressing, aggregate conditions:*
 - Space Environment
 - Launch Environment
 - **Designed Environment → This is where software “lives”**
 - Operational Environment

**This is an experience-based recommendation; unfortunately, [DoD 2009] does not have adequate space-related guidance.*

Rating CTE Maturity Using the TRL “Thermometer”



Before We Move On...



- Check your understanding of the following new terms

– *CTE*



– *IPA*



– *IPAT*



– *IRT*



– *TRA*



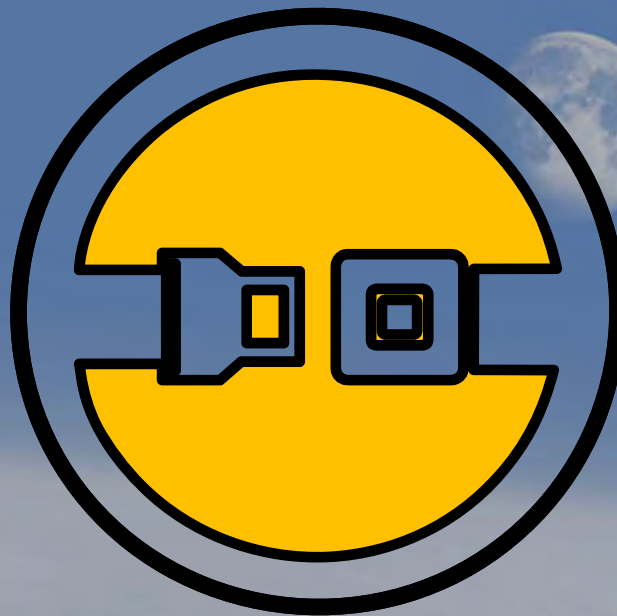
– *TRL*



– *Relevant Environment*



However, the high-altitude cruising is over...



Fasten your seat-belt and prepare for
landing!

Tutorial Scope

- Tutorial scope
 - *The TRA is an ambiguous and controversial process, but making policy-change recommendations is out of scope for this tutorial*
 - *Staying close to the objective stated in the title, we will only discuss those problems, which we believe can be mitigated via the use of software architecture standards*
- Specific issues that we will discuss
 - *Selected deficiencies of the software critical technology element (CTE) identification process*
 - *Selected deficiencies of the software technology readiness level (TRL) determination process*
 - *Why focus on architecture*
 - *Why the emphasis on standards*

Risks of Software CTE Identification

- **Overestimation:** Too many CTEs are identified
 - *The consequence of this risk is that the evaluation process becomes lengthy and expensive*
- **Underestimation:** Some truly critical technology elements are missed
 - *The consequence of this risk is cost and schedule exposure during development and manufacturing*
 - *Independent evaluations* show that underestimation is the more prevalent and pervasive risk*
- The following risks will be addressed:
 - *Missing definitions of **software technology** and **software technology element***
 - *Inadequate sources for software CTE identification*
 - *Adversarial relationship between the Government and Contractor*

* See [GAO 2005] and [DAPA 2006]

Missing TRA Definitions



- The definitions for software technology and *software technology element* are missing from the DoD TRA Deskbook*
 - *The process owners' assumption is that these concepts are trivial ("everybody knows what they mean...") and there is no need to define them*
 - *Other "guidance" on the subject is to "go and look it up in a dictionary". However, the generic, dictionary definitions are inadequate*
- It is true that on the basis of common dictionary definitions "software" itself qualifies as a "technology"
 - *However, by misunderstanding and misusing this statement, many people believe that the software that is being developed is the technology that needs to be evaluated*
 - *This misconception leads to the conflation of software technology maturity and software product maturity, and the confusion between software technology risks vs. software technical and programmatic risks*

To conduct meaningful software technology maturity assessments a further refinement and breakdown of the definition are needed

* *Actually these definitions are missing for hardware as well...*

Proposed Definition of Software Technology



- The Definition of Software Technology
 - *Software technology is defined as the theory and practice of various sciences applied to software development, operation, understanding, and maintenance. Software Technology is any concept, process, method, algorithm, or tool whose primary purpose is the development, operation, understanding, and maintenance of software*
 - Original source is [Foreman 1997,] but also adopted by [USAF 2009]
 - *Software technology examples*
 - Technology directly used in the objective system
 - *E.g., two-tier and three-tier architectures, Service-Oriented Architecture (SOA,) Remote Procedure Calls (RPC)*
 - Technology used in tools that produce or maintain software
 - *E.g., Graphical User Interface (GUI) builders, programming languages and compilers, cyclomatic complexity analyzers*
 - Process technologies applied to produce or maintain software
 - *Personal Software Process (PSPSM), Cleanroom Software Engineering*

SM PSP is a service mark of Carnegie Mellon University

SOA – A Software Technology Example

- Net Centricity – a typical, new mission requirement
 - *Network Centric Warfare (NCW)*
 - NCW is a state-of-the art war-fighting theory with the following two implementation dimensions
 - *Network Centric Operations (NCO), dealing with the cognitive and social dimensions of NCW*
 - *Network Centered Infrastructure (NCI), addressing physical and information dimensions of NCW*
 - NCI represents a new complexity concern for us because, almost automatically, puts every weapon system in a System of Systems (SoS) context
- Service-Oriented Architecture (SOA)
 - *SOA is an emerging architecture style that may be used to implement NCI. Note that NCI is strongly promoted* by the Office of the Undersecretary of Defense for Acquisition, Technology, and Logistics (OUSD(AT&L))*
 - *Note that the use of SOA has far-reaching Information Assurance and Security Technology implications*

* Source: [OUSD 2008], also see “Net Ready” as a standard Key Performance Parameter (KPP) in [DoD 2008]

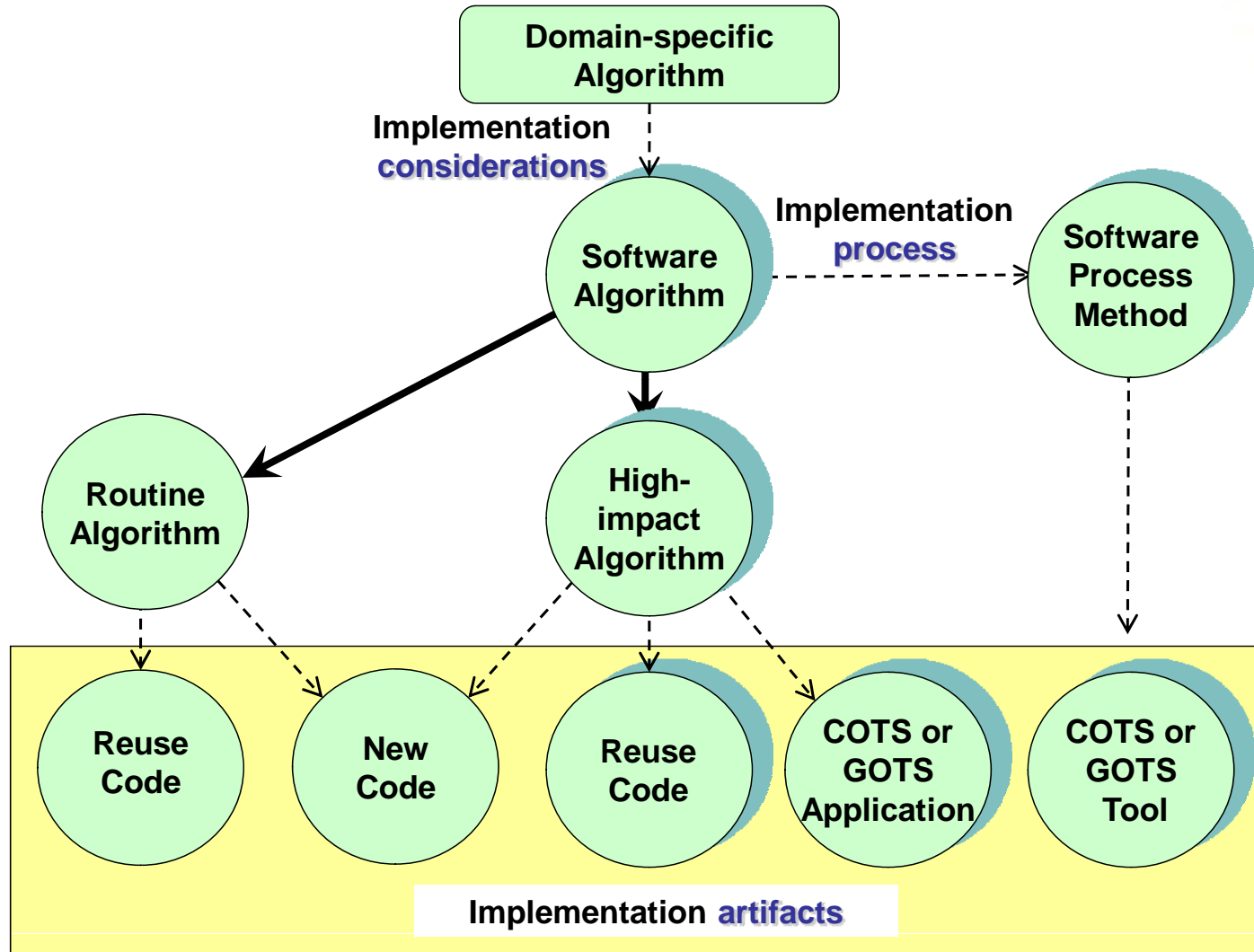
Algorithms



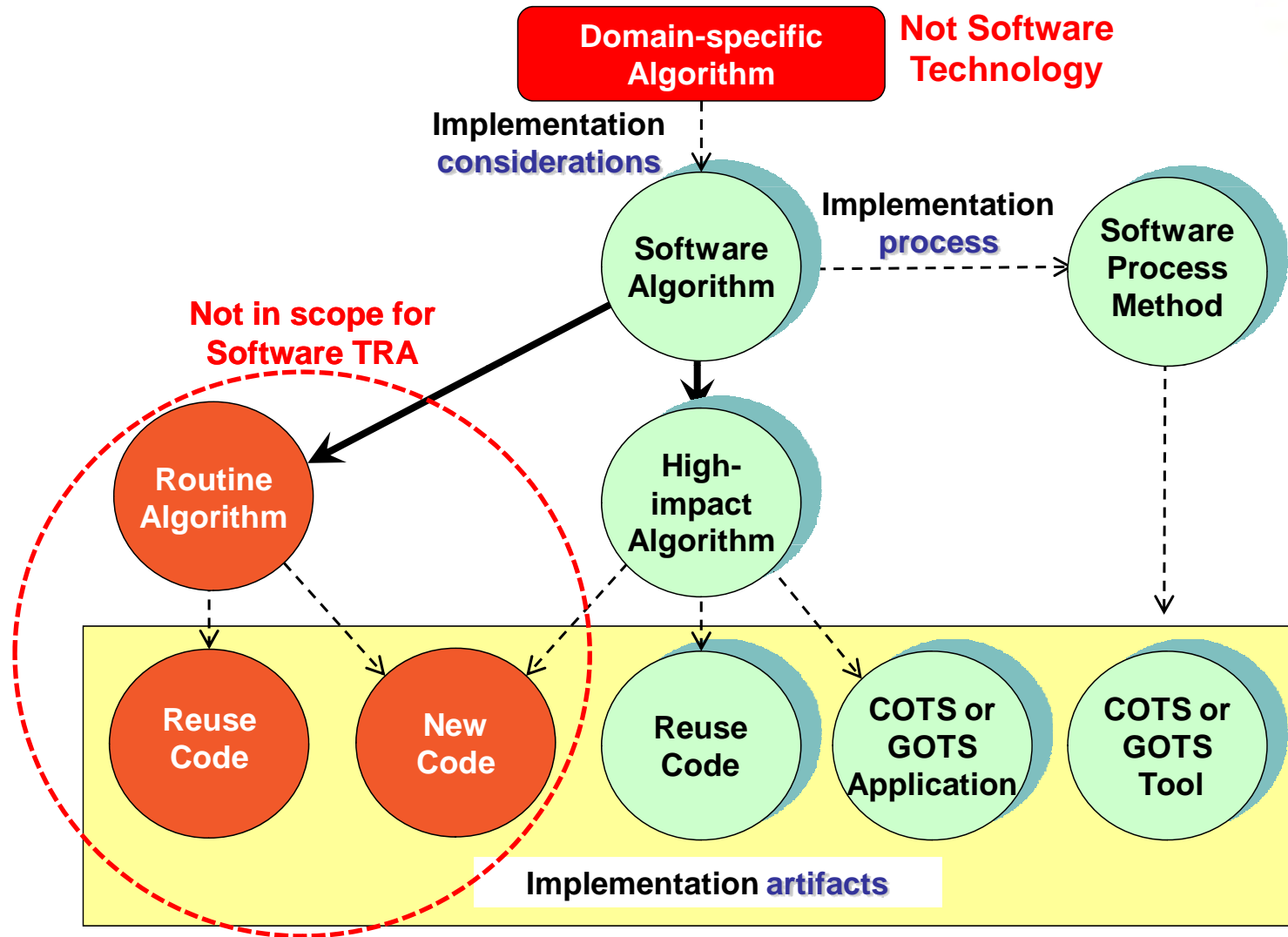
- Basic, conventional definition
 - *An algorithm is a sequence of finite instructions. It is formally a type of effective method in which a list of well-defined instructions will, when given an initial state, proceed through a well-defined series of successive states, eventually terminating in an end-state.*
- Classification of algorithms from a TRA perspective*
 - *Domain-specific algorithms*
 - Domain-specific algorithms implement various tasks in the user's domain
 - *Software algorithms*
 - Software algorithms implement various tasks in the software development domain
 - Implementation variations for software algorithms
 - *New code*
 - *Reuse code*
 - *Commercial Off-the-Shelf (COTS) and Government Off-the-Shelf (GOTS) software*

**Some source of confusion is that domain-specific algorithms might be implemented in hardware, firmware, or software. Also, domain-specific algorithms are often tested with software tools, but that does not make them software algorithms.*

Implementation Considerations for Algorithms



What is In-scope for a Software TRA?



Proposed Definition of a Software Technology Element



- Element*
 - *A fundamental, essential, or irreducible constituent of a composite entity*
- Software Technology Element (new, proposed definition)
 - *The included definition is based on the above, generic dictionary definition*
 - **Software Technology Element is a fundamental, essential, or irreducible constituent of the associated software technology, declared in the specific context of the application of said technology**
 - *“Specific context” needs to be interpreted in the following two dimensions*
 - Focus/limitations on the aspects of software technology that are relevant to the objective system
 - Focus/limitations on the partitions of the objective system’s architecture that are directly affected by said technology
 - *Nevertheless, the maturity of a particular technology element is an emerging concept and during maturity evaluation not only the directly affected architectural partition but a greater architectural context needs to be considered (Also referred to as “integration” or “integrability” of technologies)*

26 * Source: [Houghton 2009]



Using the Department of Defense Architecture Framework (DoDAF) Version 2.0 to Identify CTEs

Department of Defense Architecture Framework (DoDAF) Version 2.0*

- DoDAF is DoD's architecture framework, defining a standard approach for describing, presenting, and integrating DoD architectures. It covers both warfighting and business operations and applies to all DoD components.
 - *All major DoD weapons and information technology (IT) system acquisitions are required to develop and document an enterprise architecture using the views prescribed in DoDAF*
 - *DoDAF is also a reference model to organize the enterprise architecture and the systems architecture into complementary and consistent views*
 - *DoDAF is meant to be suited to describe large systems with complex integration and interoperability challenges*
- Architectural data representation in DoDAF
 - *Architectural Description → Viewpoints → Views → Models*

* Source: [DoDAF 2009]

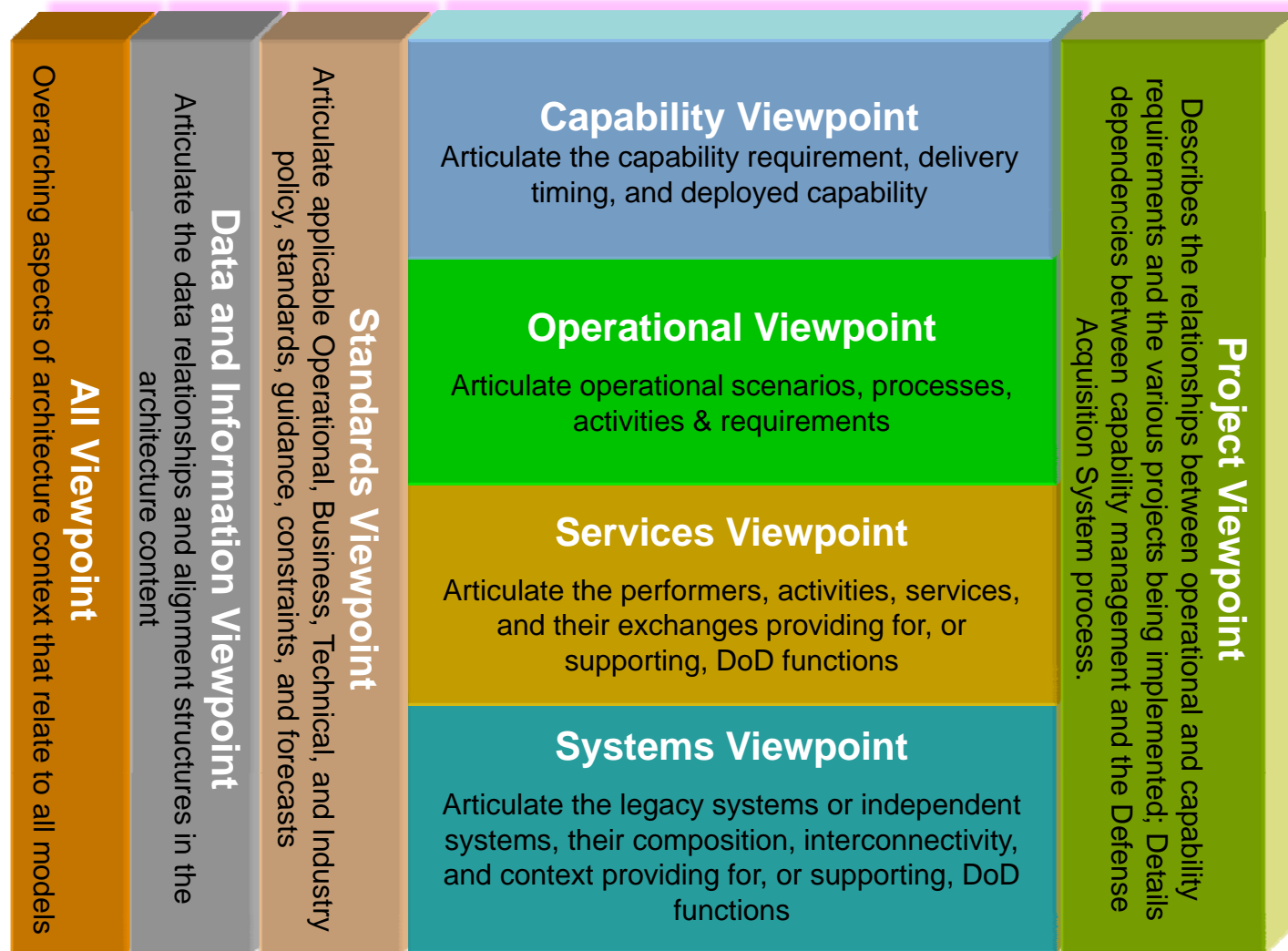
Model Types in DoDAF Version 2.0

- **Tabular**
 - *Models which present data arranged in rows and columns, which includes structured text as a special case*
- **Structural**
 - *This category comprises diagrams describing the structural aspects of an architecture*
- **Behavioral**
 - *This category comprises diagrams describing the behavioral aspects of an architecture*
- **Mapping**
 - *These models provide matrix (or similar) mapping between two different types of information*
- **Ontology**
 - *Models which extend the DoDAF ontology for a particular architecture*
- **Pictorial**
 - *This category is for free-form pictures*
- **Timeline**
 - *This category comprises diagrams describing the programmatic aspects of an architecture*

DoDAF Version 2.0 Key Definitions

- Models
 - *Created from the subset of architectural data for a particular purpose using model types as templates*
 - *Note that only the appropriate model types need to be instantiated*
- Views
 - *A view is a presentation of a portion of the architectural data*
 - ***DoDAF does not prescribe any particular views***, but instead concentrates on data as the necessary ingredient for architecture development
 - However, other regulations and instructions from both DoD and the Chairman of the Joint Chiefs of Staff (CJCS) may have particular presentation view requirements
 - *Views are meant to be “Fit-for-Purpose”, i.e., user-defined and created for a specific purpose*
- Viewpoints
 - *A DoDAF Viewpoint is a selected set of architectural data that has been organized to facilitate visualization in an understandable way*

DoDAF Version 2.0 Viewpoints*



* Source: [DoDAF 2009] Unfortunately, detailed discussion is out of scope

A TRL-focused Analysis of DoDAF

- We want to determine to what extent would DoDAF facilitate the identification of software critical technology elements
- Viewpoints to consider with technology-related information
 - *Systems Viewpoint*
 - SV-9 Systems Technology and Skills Forecast
 - *The emerging technologies, software/hardware products, and skills that are expected to be available in a given set of time frames and that will affect future system development*
 - *Services Viewpoint*
 - SvcV-9 Services Technology & Skills Forecast
 - *The emerging technologies, software/hardware products, and skills that are expected to be available in a given set of time frames and that will affect future service development*
 - *Standards Viewpoint*
 - StdV-1 Standards profile
 - *The listing of standards that apply to solution elements*
 - StdV-2 Standards forecast
 - *The description of emerging standards and potential impact on current solution elements, within a set of time frames*

Evaluation

- Overall
 - *DoDAF Viewpoints may be the source of some relevant information; however, the whole framework is focused on enterprise-level modeling and as such, too high level for discovering software technology elements*
- Standards Viewpoint
 - *Most state-of-the-art technologies are not covered by any standards*
 - In fact, standards could slow down or might paralyze the competition of technology vendors
- Systems and Services Viewpoints
 - *Both viewpoints require the forecasting of technologies*
 - *However, neither viewpoint requires the documentation of currently planned technologies at any levels*
 - *Technology forecasts are useful to evaluate the extensibility or robustness of the system but insufficient to the evaluation of the current technology solutions*


Conclusion: DoDAF is inadequate to facilitate software CTE Identification

Final Caution Regarding Older Versions of DoDAF



- During the migration from Version 1.5 to Version 2.0 DoDAF was substantially overhauled and restructured
- As an effort to align DoDAF with other, prevailing architecture description standards, several, basic terms were redefined

DoDAF V1.5	DoDAF V2.0
Architecture	Architectural Description
Architecture data	Architectural data
Product	Model (a template for collecting data)
Product	View (a model with data for an architecture)
View	Viewpoint



Using the Work Breakdown Structure (WBS) to Identify CTEs

The WBS is Inadequate for CTE Identification

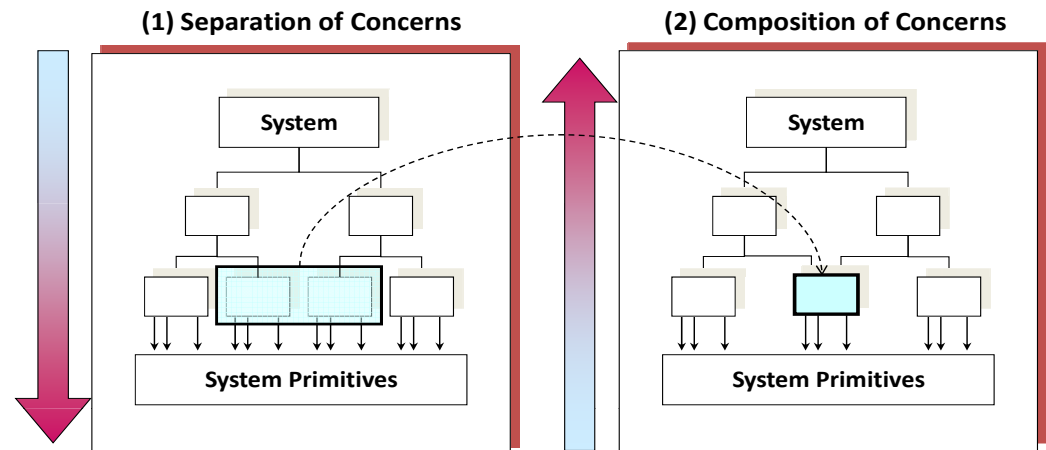
Level	WBS Description
1	Space System
2	Systems Engineering, Integration, and Test/Program Management and Common Elements
2	Space Vehicle (1...n as required)
3	Systems Engineering, Integration, and Test/Program Management and Common Elements
3	Spacecraft Bus
4	Systems Engineering, Integration, and Test/Program Management and Common Elements
4	Structures and Mechanisms Subsystem
4	Thermal Control Subsystem
4	Electrical Power Subsystem
4	Attitude Control Subsystem
4	Propulsion Subsystem
4	Navigation Subsystem
4	Spacecraft Bus Control
5	Spacecraft Bus Processor Hardware
5	Spacecraft Flight Software
6	Spacecraft Flight Software Build 1...k
3	Communication
4	Systems Engineering, Integration, and Test/Program Management and Common Elements
4	Communication Hardware (1...n as required)
4	Communication Flight Software (1...n as required)
5	Communication Flight Software Build 1...k
3	Payload
4	Systems Engineering, Integration, and Test/Program Management and Common Elements
4	Payload Hardware (1...n as required)
4	Payload Flight Software (1...n as required)
5	Payload Flight Software Build 1...k
3	Booster Adapter
3	Space Vehicle Storage
3	Launch System Integration
3	Launch Operations and Mission Support
2	Ground (1...n) as required
3	...
2	Launch Vehicle

Space example, based on MIL-HDBK-881A, Appendix F

Software only shows up at the 5-6th levels and its designation lacks details

More Concerns Regarding the Use of the WBS

- The WBS only represents a functional, hierarchical decomposition of requirements
 - *This approach is outdated and not typical in current software development*



- The WBS only represents the “What” – no help with the “How”
 - *However, technology is about the “How”, i.e., implementation (e.g., COTS)*
- Similarly, the WBS does not have features to show
 - *The software development environment’s tools*
 - *Where and how process technologies are used*
- When only WBS is used, several aspects of the system stay unclear
 - *Architectural details and component dependencies*
 - *Architectural decisions with potential technology impact*
 - *Technology concerns requiring architectural changes*

Software architectural description can mitigate most of these concerns



Exploring the Use of the Architecture Description Standards to Identify CTEs

The ISO/IEC 42010:2007 Architecture Standard

- ISO/IEC 42010:2007, Recommended Practice for Architectural Description of Software-intensive Systems
 - *ISO/IEC 42010:2007 is the equivalent of IEEE Std 1471-2000, IEEE Recommended Practice for Architectural Description of Software-intensive Systems*
 - *It is a conceptual framework to address the activities of the creation, analysis, and sustainment of architectures of software-intensive systems, and the recording of such architectures in terms of architectural descriptions*
 - *In its informative annexes heavily references ISO/IEC 10746, Information Technology – Open Distributed Processing Reference Model*
- Architectural data representation
 - *ISO/IEC 42010:2007*
 - Architecture description → Views → Viewpoints → Viewpoint Language
 - *ISO/IEC 10746*
 - Architecture description → Viewpoints → Viewpoint Language
 - *Note the lack of View definition in ISO/IEC 10746*

ISO/IEC 42010:2007 Terminology

- View

- *An architecture description is organized into one or more constituents called (architectural)views. A view is a representation of a whole system from the perspective of a related set of concerns*

- Viewpoint

- *A specification of the conventions for constructing and using a view. A viewpoint is a pattern or template from which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis*

- Viewpoint ↔ View relationship

- *A viewpoint is to a view as a class is to an object in object-oriented programming where “class” is a template and “object” is an instance*
 - viewpoint : view :: class : object

- Viewpoint Language

- *Defines “the concepts and rules for specifying systems from the corresponding viewpoint*

Selected ISO/IEC 42010:2007 Viewpoints

- **Structural Viewpoint**
 - *Elements, components of the system*
 - *Interactions between these components (“connectors”)*
 - *Structural organization of elements*
- **Behavioral Viewpoint**
 - *Dynamic actions of and within the system*
 - *Actions produced by the system*
 - *The ordering and synchronization of these actions*
 - *The behavior of system components and their interactions*
- **Physical Interconnect Viewpoint**
 - *Physical communication interconnects among system components*
 - *Layering among system components*
 - *Feasibility of construction, compliance with standards, and evolvability*

Note that all this information is relevant to technology selection and evaluation but even an elaborate WBS would not show it

Environmental Information Needed for Determining Relevant Environment for Software

- Environmental categories according to the DoD TRA Deskbook:
 - *External or imposed environment*
 - Related to the operation of the product, may be either natural or man-made
 - *Internal or designed environment*
 - Always man-made, related to the designed product
- Further environmental dimensions
 - *Physical environment (for software, it is the designed environment)*
 - *Logical environment*
 - *Data environment*
 - *Security environment*
 - *User and use environment*

Software architectural description can provide most of the needed information

What About the Technology Viewpoint in ISO/IEC 42010?

- Basic goals (concerns) are exactly what we are looking for
 - *Capturing the choice of technology in the system*
 - *How specifications are implemented*
 - *Specification of relevant technologies*
 - *Support for testing (verification)*
- Viewpoint language
 - *Unfortunately, no direct guidance on how to create a Technology View for a system*
 - *The only - informative - reference is to ISO/IEC 10746*

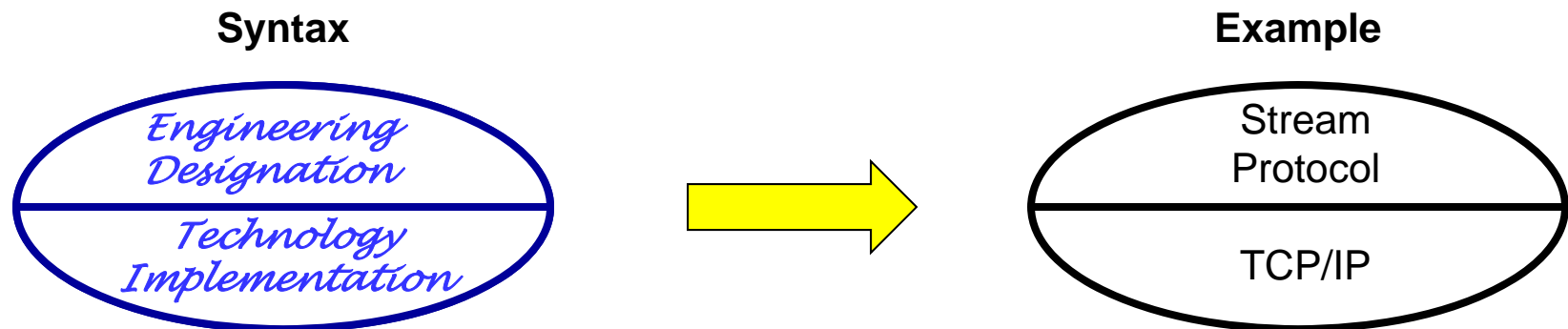
Adversarial Relationship Between the Government and Contractor

- Technology Readiness Assessment is a policy-driven acquisition process, conducted by the Government's independent review panel
 - *The contractors provide initial assessment and supporting information*
- However, the Government and the Contractor have conflicting interests
 - *While the Contractor is certainly interested in understanding the technology risks of the project, its main goal is winning the contract, which can only be achieved via minimizing the projected risks of their technology solution and providing a low-cost bid*
 - *On the other hand, the Government's interest is not to get engaged in acquisitions with high technology risks; see the applicable Public Law*
- Currently the TRA is a laborious discovery process and its success highly depends on contractor support and the time an independent review panel can spend on the assessment

New ideas needed - None of the discussed methods can facilitate the explicit declaration of technology solutions

Exploring ISO/IEC 10746

- ISO/IEC 10746, Information Technology — Open Distributed Processing - Reference Model
 - *This is a very extensive and specialized standard, providing a framework for creating an architecture within which support of distribution, internetworking, interoperability, and portability can be integrated for open distributed systems*
- Technology Viewpoint in ISO/IEC 10746
 - *Technology Viewpoint is the means to specify technologies in a system*
 - *The system is visible in the technology viewpoint in terms of statements by the supplier, such as choices of specific hardware and software components compliant with the other viewpoint specifications*



This is the approach we would like to exploit

Unified Modeling Language (UML®)

- What is it?
 - *UML is a general-purpose visual modeling language that is used to specify, visualize, construct, and document the artifacts of a software system [Rumbaugh 1999]. Current release is Version 2.0*
 - *It is a successor to numerous object-oriented analysis and design methods of the late '80s and early '90s*
 - *Since 1998 UML is a standard adopted and maintained by The Object Management Group (OMG)*
 - *Note that while UML was historically based on object-oriented methods, it does not assume or require the use of any particular development method. In fact, it is applicable in case of non object-oriented methods as well.*
 - *UML has a rich syntax and broad spectrum of modeling constructs*
- Architectural data representation in UML
 - *Architecture description → Views → Diagrams (Modeling constructs)*

A view is a subset of diagrams representing one aspect of a system

® UML is registered in the U.S. Patent and Trademark Office by The OMG

UML Views and Diagrams

- Views

- *Static*
- *Use Case*
- *Implementation*
- *Deployment*
- *State Machine*
- *Activity*
- *Interaction*
- *Model Management*

- Diagrams

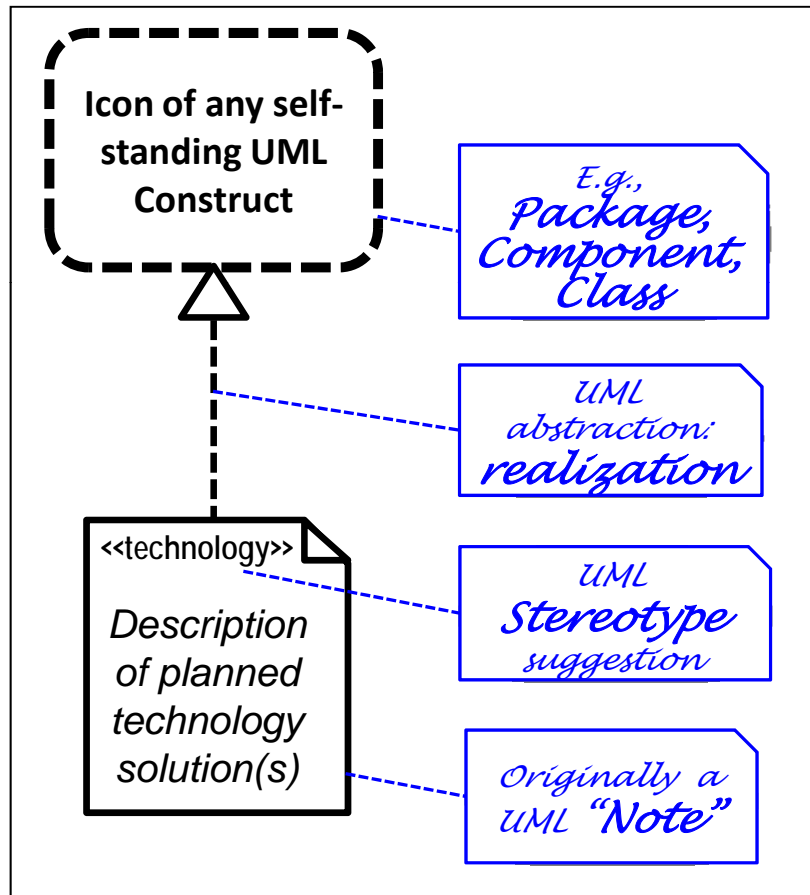
- *Class*
- *Use Case*
- *Component*
- *Deployment*
- *Statechart*
- *Activity*
- *Sequence*
- *Collaboration*

UML is a promising approach as a Viewpoint Language for ISO/IEC 42010

Proposed UML Specification of the ISO/IEC 42010 Technology Viewpoint



Syntax



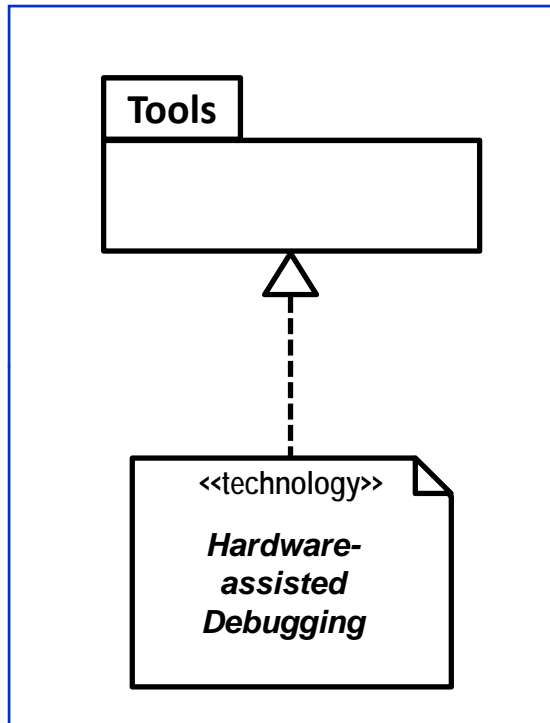
Semantics

- The planned software technology solutions should fall into one or more of the earlier mentioned categories:
 - *High-Impact Software Algorithm*
 - Reuse Code Implementation
 - COTS/GOTS Implementation
 - *Software Process Method*
 - *COTS/GOTS Tools*
- Technology Element clarification
 - *The UML notation automatically provides the architectural focus*
 - *The expectation for the description is to provide further limitations as they apply to the objective system*

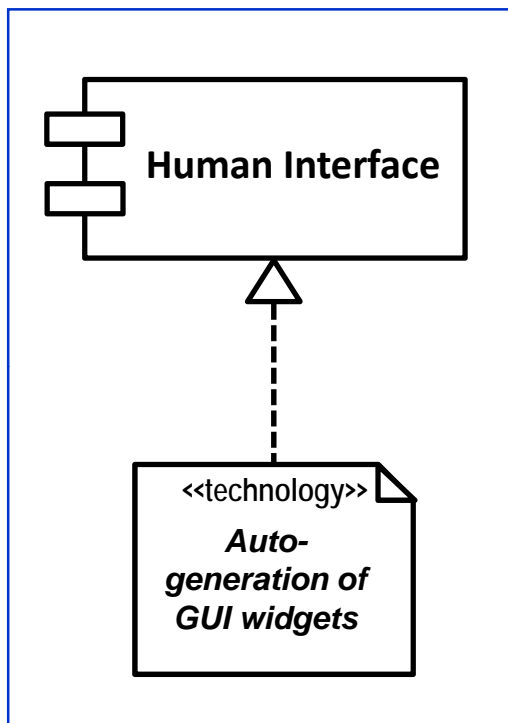
Legend: Syntax: Rules to manipulate symbols based on their shapes;
Semantics: What the symbols mean

Examples

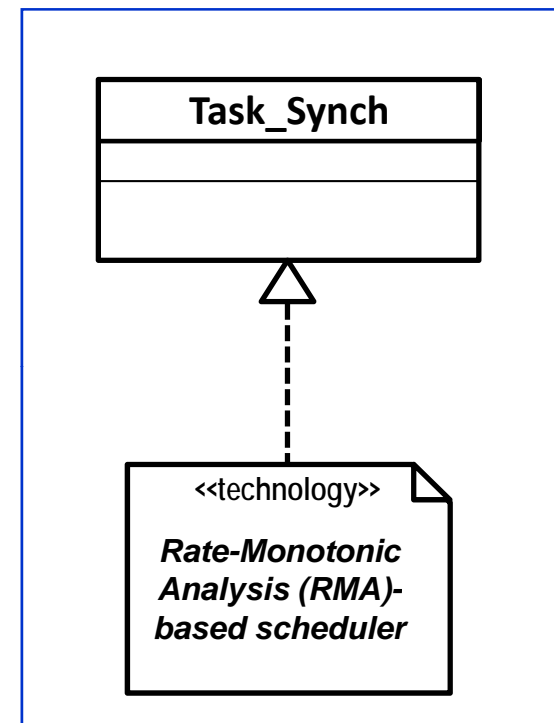
Packet



Component



Class



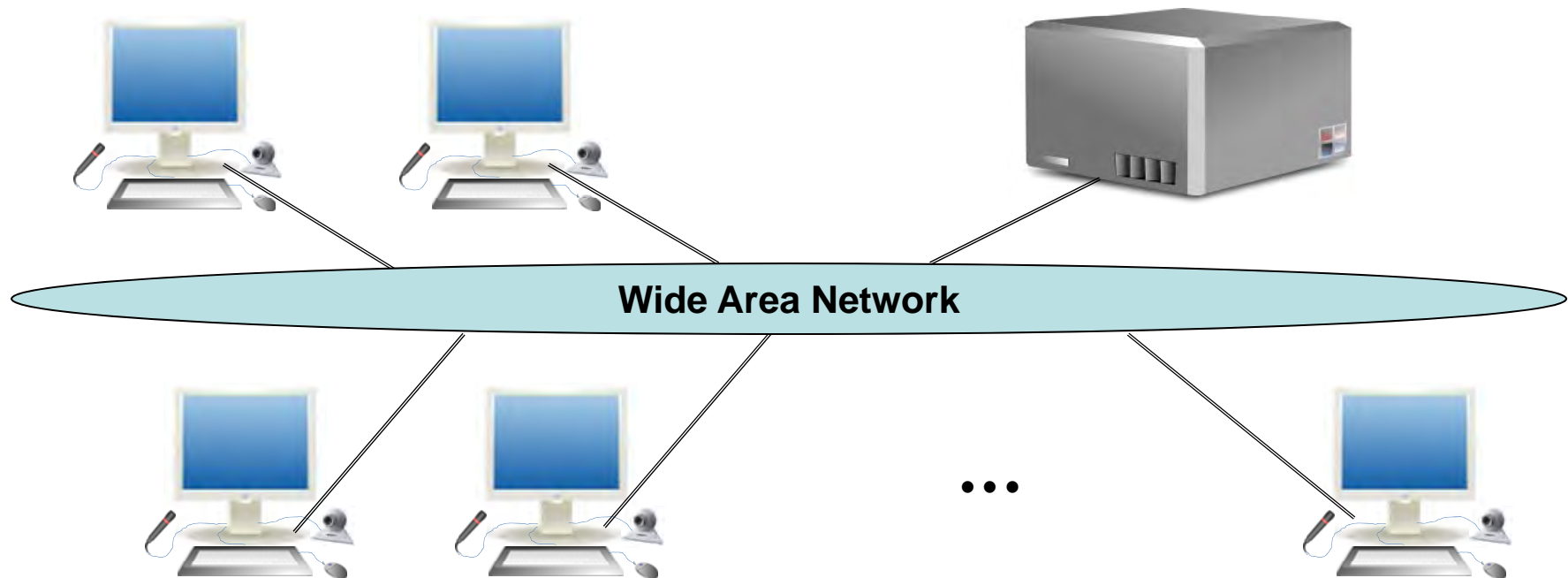
- Caveats
 - *Architecture, and consequently its views, are not static entities; architectural details and depth gradually evolve during development*
 - *As a result, such details depend on the positioning of our assessment in the system development life cycle*
 - *Our ability to identify CTEs is a function of the level and resolution of the architectural description at the time of the assessment*

Case Study

Multimedia Conferencing System – Technology Specification

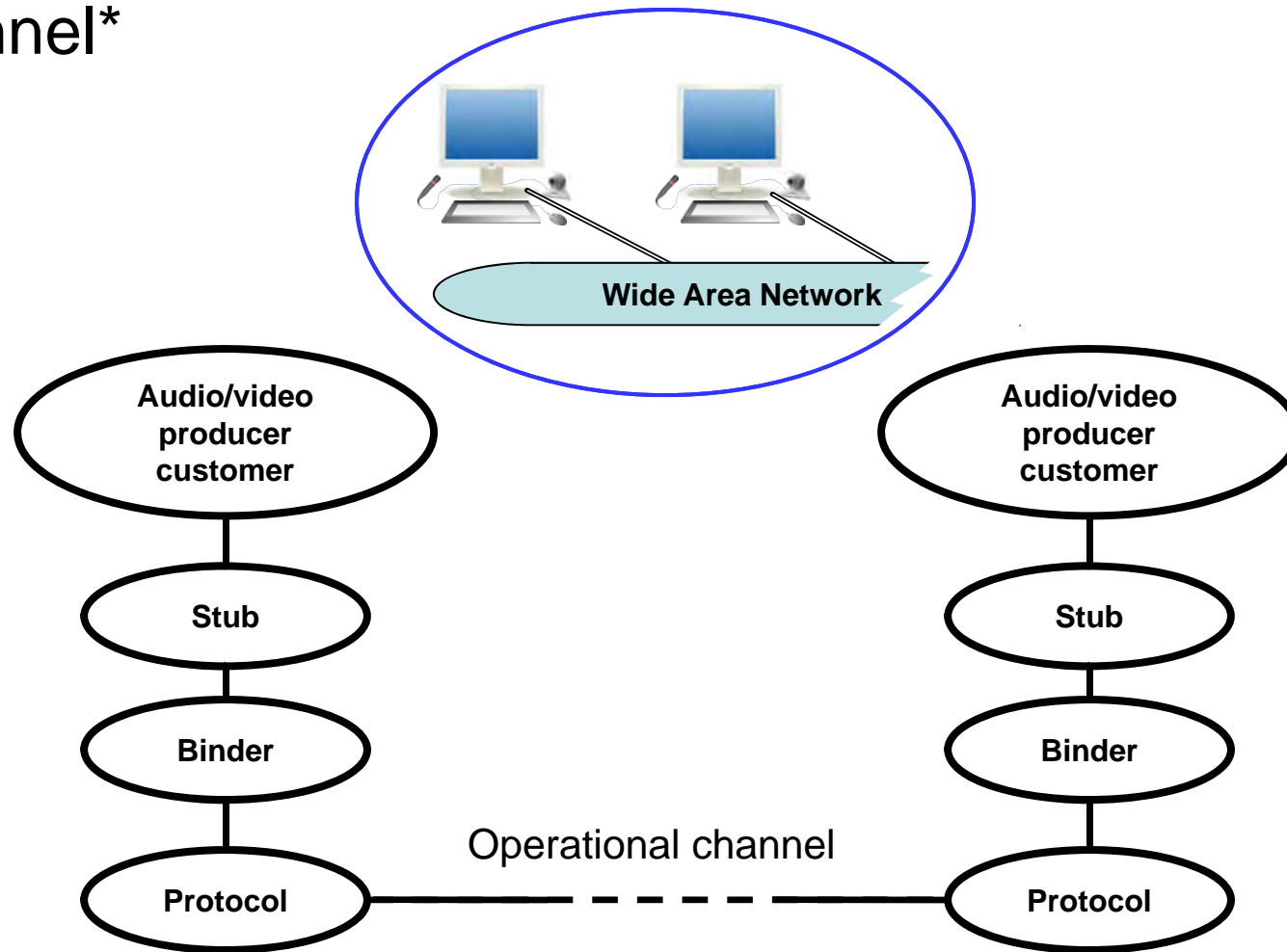
Multimedia Conferencing System*

- The Multimedia Conferencing System (MMCS) allows real-time interworking between several users using multimedia information
 - *The service enables a group of persons that are physically distributed to work together on multimedia information and communicate with one another*
 - *Multimedia information consists of text, electronic mail and audio/video streams*



* Based on [ISO/IEC 1996], paragraph 12.1, page 53

Partial Engineering Specification of an Operational Channel*



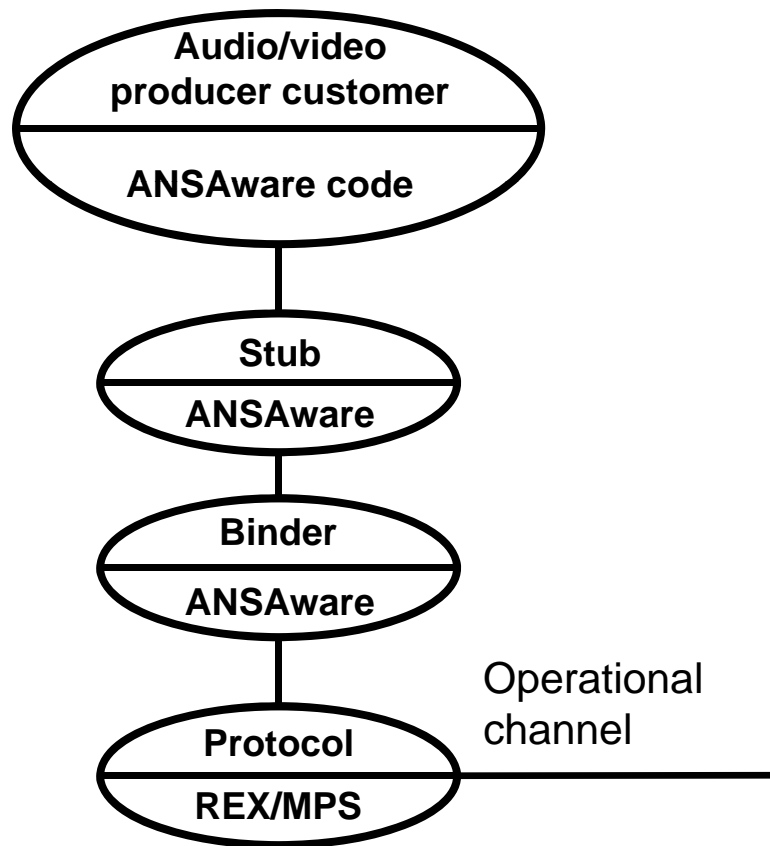
Operational channel creation between two audio/video producer customers

* Based on: [ISO/IEC 1996], Figure 35, page 67

Engineering to Technology Mapping in ISO/IEC 10746*

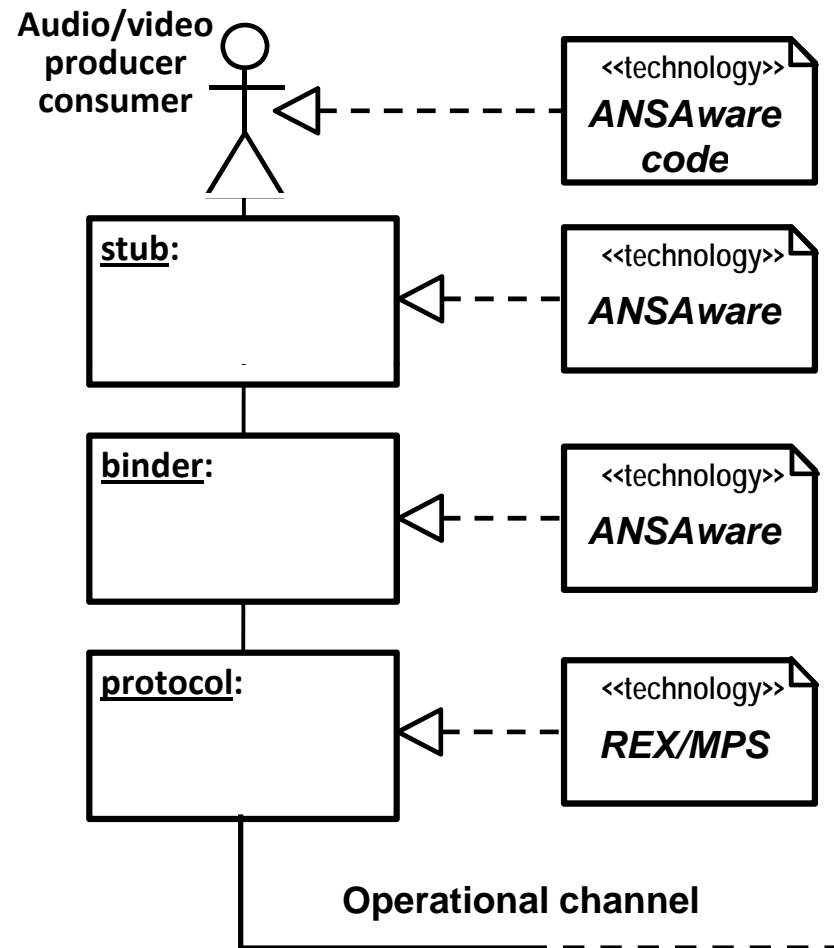
- Legend

- **Stub**: A stub object provides adaptation functions to support interaction between basic engineering object interfaces in different nodes
- **Binder**: Binders are responsible for validating the interface reference and maintaining the integrity of the binding
- **Protocol**: The protocol object assures that computational objects can interact remotely with each other
- **ANSA**: (Advanced Network System Architecture) was a U.K. industrial consortium, managed by Architecture Project Management Limited. Ceased to exist in 1998
- **ANSAware**: It used to be an ANSA product - a software suite to realize a distributed and networked system
- **REX**: Remote Execution Protocol
- **MPS**: Message Passing Service



* Based on: [ISO/IEC 1996], Figure 36, page 69

UML Implementation of the Engineering to Technology Mapping*



* Instead of using the earlier and unique, split oval icons, a standard UML object model is shown where the object descriptions are tagged with "technology"-stereotyped note icons

Discussion of the Case Study

- First some UML peculiarities...
 - *It seems odd that the “stick man” figure (a UML actor) has a technology tag*
 - However, in UML, “actor” is an abstraction for describing entities that interact with a system or classifier. The “stick man” is a stereotyped icon introduced for such entities and may represent nonhuman actors such as systems, subsystems, classes, or processes as well
 - *Caveat: Oval icons that were used in ISO/IEC 10746 to depict objects, are reserved for use cases in UML*
 - This is kind of an unfortunate after-effect of the “unification” effort that drove the development of UML
- In the Case Study all objects have only one associated technology
 - *This is only the case because the original example in ISO/IEC 10746 was presented this way and we wanted to maintain the symmetry*
 - *However, in the scheme we are proposing, any of the UML constructs may have multiple technologies associated with it*
 - For example, high-impact software algorithm, COTS realizing the algorithm, and a specialized software tool to debug the software

Discussion of the Case Study (cont.)



- Would you identify any of the technologies (technology elements) of this example as “critical”?
 - *Case a: Only the engineering solution (i.e., object diagram, without the technology tags) is presented – this is the classic, CTE “discovery” mode*
 - Creating an operational channel between audio/video producers via a network does not seem to be either a new or difficult problem requiring unique technology solutions ; hence the solution would be probably classified as non-critical and would not be subjected to further technology readiness assessment and rating
 - *Case b: Technology tags are shown as well*
 - None of the technologies are particularly new or novel and they are quite mature (the evidence is that they made it into an ISO/IEC standard in 1996.) On this basis they would be classified as non-critical
- However
 - *The problem with these, proposed solutions is not lack of maturity but being obsolete, considering that the ANSA consortium ceased to exist in 1998(!)*
 - ***On that basis a red flag needs to be raised early about the associated risks and infeasibility of the proposed solution!***

Risks of Software TRL Determination

Risks of Software TRL Determination

- TRLs are determined in isolated “silos”
 - *TRLs are to be reported independently for all involved technical disciplines (technical domains)*
- Conducting software TRAs was an afterthought
 - *The TRA was originally a hardware assessment, providing details for all technology areas (e.g., propulsion, antennas, battery, etc. for a space system)*
 - *“Software” was added after the recognition that most acquired weapon systems are software-intensive and software plays a definitive role*
- However, the interplay between software and hardware, primarily electronics and electro-mechanical hardware, is not well comprehended in the process

Basic TRL Definitions from the DoD TRA Deskbook

TRL	Basic Hardware TRL DEFINITIONS from the DOD TRA Deskbook	Basic Software TRL DEFINITIONS from the DOD TRA Deskbook
1	Basic principles observed and reported	Basic principles observed and reported
2	Technology concept and/or application formulated	Technology concept and/or application formulated
3	Analytical and experimental critical function and/or characteristic proof of concept	Analytical and experimental critical function and/or characteristic proof of concept
4	Component and/or breadboard validation in a laboratory environment	Module and/or subsystem validation in a laboratory environment
5	Component and/or breadboard validation in a relevant environment	Module and/or subsystem validation in a relevant environment
6	System/subsystem model or prototype demonstration in a relevant environment	Module and/or subsystem validation in a relevant end-to-end environment
7	System prototype demonstration in an operational environment	System prototype demonstration in an operational, high-fidelity environment
8	Actual system completed and qualified through test and demonstration	Actual system completed and mission qualified through test and demonstration in an operational environment
9	Actual system proven through successful mission operations	Actual system proven through successful mission-proven operational capabilities

Legend: **Yellow** – significant hw/sw differences; **Red** – References to the relevant environment

Additional Information to Consider

TRL	Basic Hardware TRL DEFINITIONS from the DOD TRA Deskbook	Basic Software TRL DEFINITIONS from the DOD TRA Deskbook	TRL GOALS	Knowledge Involved in Achieving Hardware Objectives	Knowledge Involved in Achieving Software Objectives	Systems Engineering Responsibilities
1	Basic principles observed and reported	Basic principles observed and reported	Demonstrate scientific feasibility	Natural Sciences	Computer Science	Requirements, Trade studies
2	Technology concept and/or application formulated	Technology concept and/or application formulated				
3	Analytical and experimental critical function and/or characteristic proof of concept	Analytical and experimental critical function and/or characteristic proof of concept				
4	Component and/or breadboard validation in a laboratory environment	Module and/or subsystem validation in a laboratory environment	Demonstrate engineering feasibility	Hardware Engineering	Software Engineering	In-domain integration Cross-domain evaluation
5	Component and/or breadboard validation in a relevant environment	Module and/or subsystem validation in a relevant environment		Systems Engineering	Systems Engineering	
6	System/subsystem model or prototype demonstration in a relevant environment	Module and/or subsystem validation in a relevant end-to-end environment				
7	System prototype demonstration in an operational environment	System prototype demonstration in an operational, high-fidelity environment	Demonstrate operational feasibility	Hardware Engineering	Software Engineering	Cross-domain integration
8	Actual system completed and qualified through test and demonstration	Actual system completed and mission qualified through test and demonstration in an operational environment		Systems Engineering	Systems Engineering	
9	Actual system proven through successful mission operations	Actual system proven through successful mission-proven operational capabilities	Demonstrate operations	Mission Domain	Mission Domain	Mission Domain Demonstration

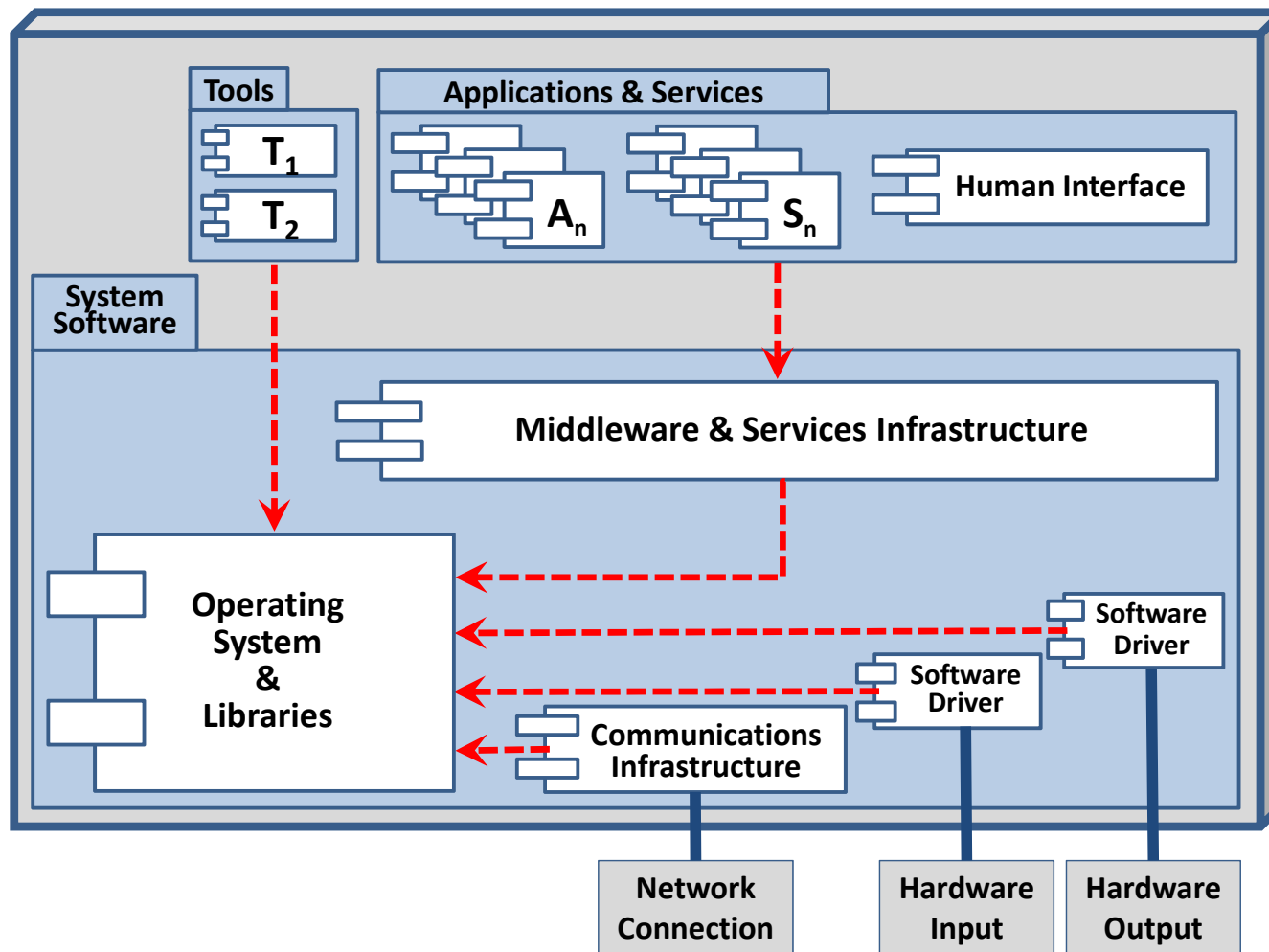
In-domain evaluation and in-domain/cross-domain integration are concerns

Exploit Available UML Dependency Information



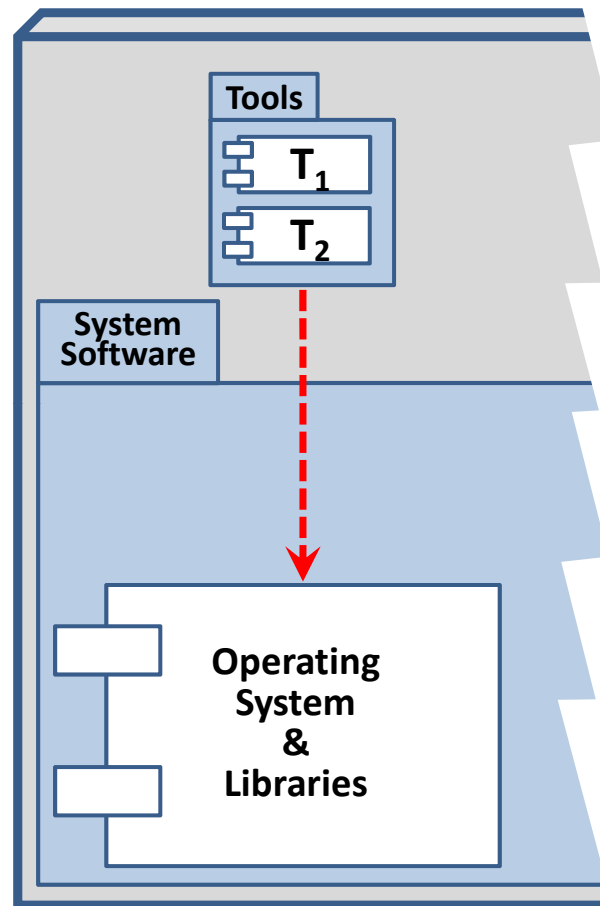
- Definition of Dependency in UML
 - *Dependencies among components show how changes to one component may cause other components to change*
 - *Type of dependencies in software-intensive systems*
 - Communication dependency
 - *Communication between software components*
 - *Communication between hardware and software components*
 - Deployment dependency
 - *Compilation dependency between software components*
 - *Dependency between software and the hosting hardware node*
- UML has appropriate constructs to depict these dependencies
 - *Finding this information does not require additional effort, i.e., if the architecture is documented in UML then the architecture models already include the relevant dependency relationships amongst the elements*
- How to exploit deployment dependency information during the TRA?
 - *If a component is associated with a CTE, then all dependent components should be considered candidate CTEs*
 - *Dependency places constraints on the components' TRL*

Example: Simplified Component/Deployment UML Diagram for a Satellite Ground System



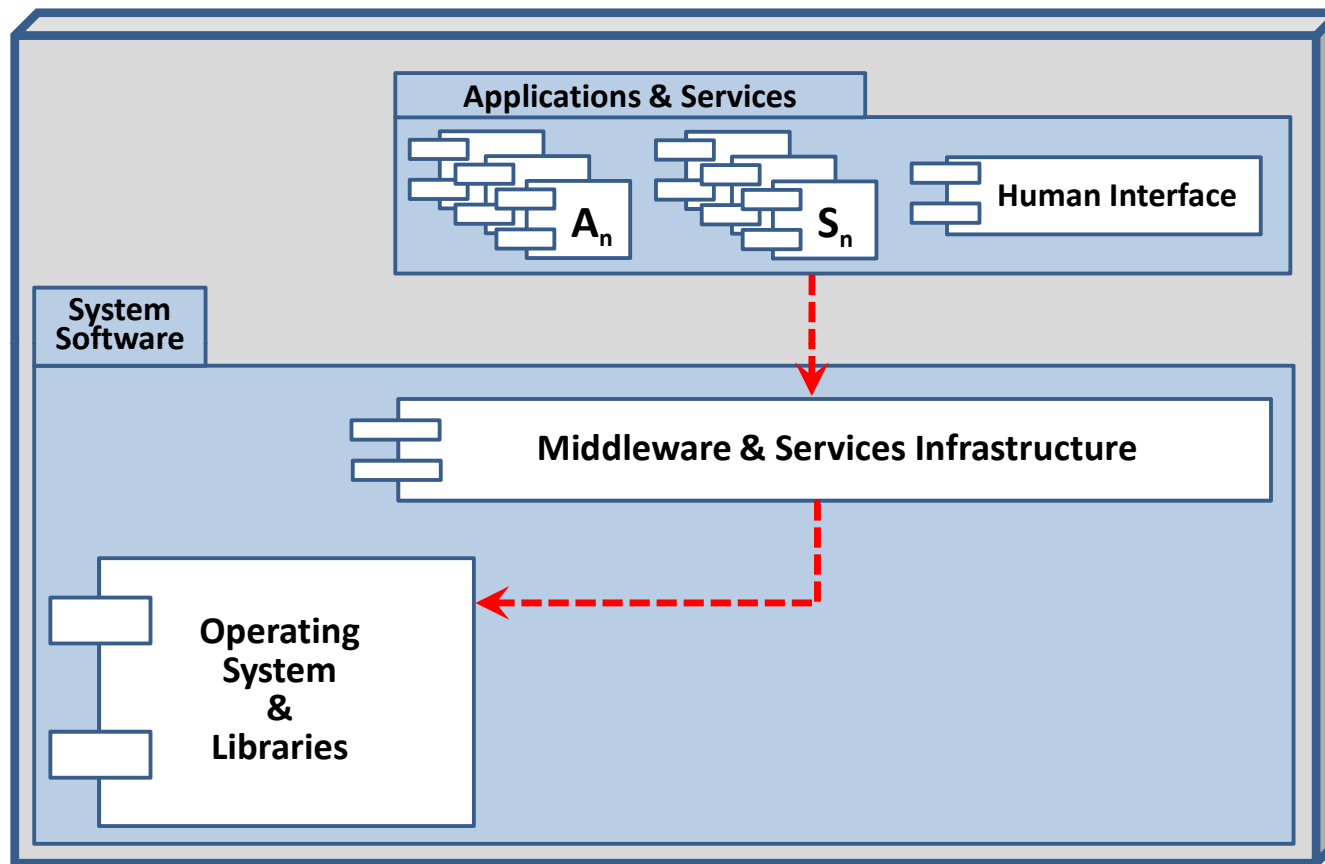
Legend: - - - - - ➔ Dependency; ————— Hardware connection

Rating-constraints for Tools Due to Dependency



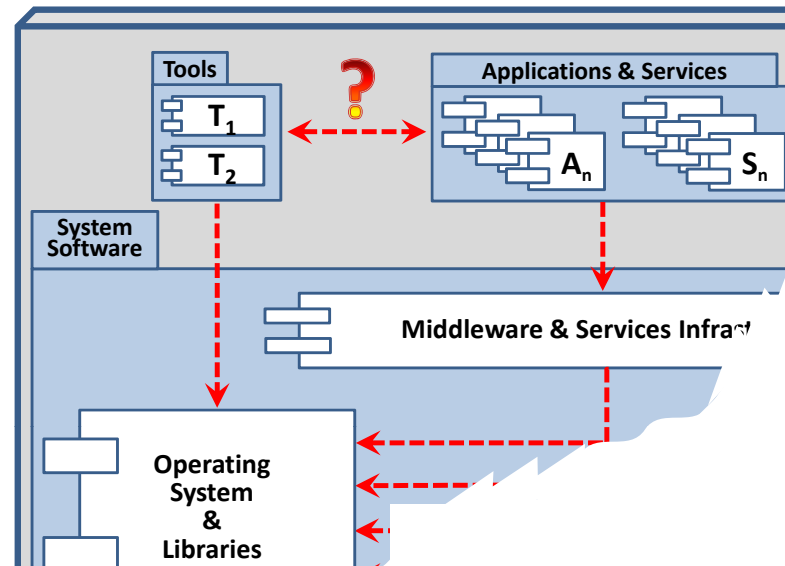
$$\text{TRL}(\text{Tools}) \leq \text{TRL}(\text{Operating System \& Libraries}) \leq \text{Hardware}$$

Rating-constraints for Applications & Services Due to Dependency



$TRL(\text{Applications \& Services}) \leq TRL(\text{System Software}) \leq \text{Hardware}$

Dependency Between Tools and Applications



- Aspects of dependency between software tools and applications
 - *Creation and debugging the code assume a synergistic relationship*
 - *Some applications might require specialized tools, although theoretically, even the most sophisticated applications can be created and debugged with rudimentary tools; hence the coupling assumed to be loose (remember core dumps? ☺)*
- Would UML facilitate the TRL rating?
 - *No, the contractors' architecture diagrams in general do not show software tools*

The analysis of this aspect of dependency is not supported directly by UML



Using Standards in the Acquisition Context

Why the Emphasis on Standards?

- Our primary focus is Mission Success [MAG 2007]
 - **Mission Success** is defined as the achievement by an acquired system (or system of systems) to singularly or in combination meet not only specified performance requirements but also the expectations of the users and operators in terms of safety, operability, suitability and supportability
 - **Mission Assurance** is the disciplined application of general systems engineering, quality, and management principles towards the goal of achieving Mission Success, and, towards this goal, this disciplined application provides confidence in its achievement
- How can it be ensured that high mission assurance processes are used to develop the objective system?
 - Use a robust development standard [Eslinger 2006]
 - Note that even the use of so-called mature processes that are based on such frameworks as the CMMI® is inadequate, and **the government must require contractual compliance with a robust development standard**
 - The foundation for this conclusion has been derived from the analysis of Acquisition Reform-induced failures on numerous space programs

® CMMI is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University

Why the Emphasis on Architecture Standards in Acquisition?

- DoDAF is already mandated
- The use of other architecture standards can facilitate the better execution of the following two DoD directives
 - *Modular Open Systems Architecture (MOSA)*
 - DoD 5000.1 requires all programs subject to a milestone review to brief the MDA on their program's MOSA implementation status
 - "MOSA" employs an integrated business and technical strategy to support the identification of the key modules and interfaces in a system's architecture
 - *Net-Ready Key Performance Parameter (NR-KPP)*
 - NR-KPP is mandated in all programs
 - The NR-KPP requires compliance with the Net-Centric Operations and Warfare (NCOW) Reference Model (RM), applicable Global Information Grid (GIG) Key Interface Profiles (KIP), DoD information assurance requirements, and miscellaneous other, supporting integrated architecture products

The Standards Dilemma



Old

- ... Requirements in solicitations are being described in performance terms
- ... Military standards cancelled

Source: "Specifications & Standards – New Way of Doing Business", June 29, 1994

Current

▶ "DoD Policy is to promote standardization of materiel, facilities, and engineering practices ..."

Source: DoD Defense Standardization Program website, <http://www.dsp.dla.mil>

Problems with Architecture Standards

- Besides the trivial one stated earlier by George F. Will, there are more, specific concerns with architecture standards
 - *There is a substantial ambiguity around essential definitions, such as the concepts of views and viewpoints*
 - *Unfortunately, full reconciliation is impossible without changing the standards*
- Summary situation
 - *DoDAF 2.0*
 - This is the least flexible standard
 - However, since we are not recommending it for software CTE identification, we do not have to deal with its definitional ambiguities
 - *UML*
 - It is used with commercial tools, which makes it somewhat inflexible
 - On the other hand, its syntax is rich and the spectrum of available modeling constructs is broad
 - *ISO/IEC 42010 and ISO/IEC 10746*
 - Lack of Views in ISO/IEC 10746 is somewhat explained in ISO/IEC 42010

The Highlights of Our Proposal Revisited

- Treat ISO/IEC 42010 definitions as primary definitions
- Use the ISO/IEC 10746 guidance for defining the ISO/IEC 42010 Technology Viewpoint
- Adopt UML as the Viewpoint Language for ISO/IEC 42010
 - *Introduce a UML stereotype called “technology” to specify the Technology Viewpoint for ISO/IEC 42010*

This proposal offers a reasonable way to deal with the ambiguities

What to Expect in the Future in the TRA Arena?

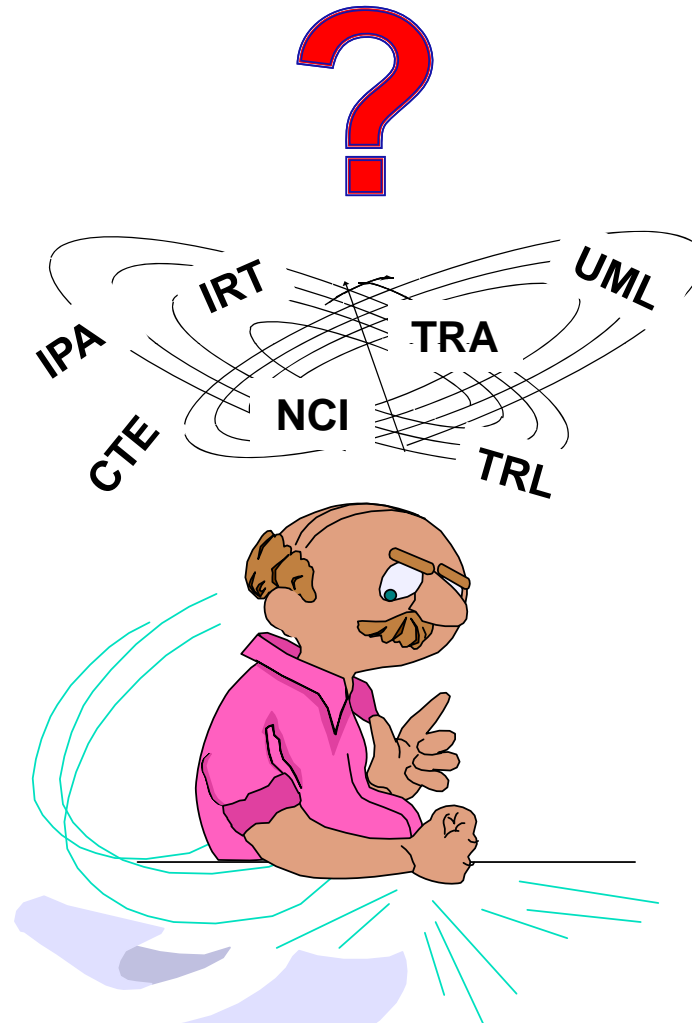
- Current State: The TRA process – as it is – ambiguous and controversial
 - *We discussed the ambiguity related to conventional risk management*
 - *Another, prevalent problem is the confusion between **technology maturity** and **product maturity***
 - Note that TRL 7-8-9 definitions are based on the characteristics of the emerging product and not the characteristics of the technology (You may revisit basic TRL definitions on slide 55)
- “Carter Memo”, September 14, 2010 [Carter 2010]
 - “Reform TRL reviews to focus on technology as opposed to engineering and integration risk. The TRL review and certification process has grown well beyond the original intent and should be reoriented to an assessment of technology maturity and risk as opposed to engineering and integration risk.”*

This directive might have far-reaching consequences in the future...

Concluding Thoughts

- Technology Readiness Assessment, a critical tool in Defense Acquisition, helps us to prevent the incorporation of immature technologies during system design
- Modern weapon systems are software-intensive systems; consequently, the early evaluation of software technologies is essential
- The approach introduced in this tutorial, based on the use of architecture standards, mitigates Critical Technology Element identification risks and also, provides substantial help during the rating of those technology elements

Questions, Comments?



Acronyms

ANSA	Advanced Network System Architecture
APO	Acquisition Program Office
AT&L	Acquisition, Technology, & Logistics
CJCS	Chairman of the Joint Chiefs of Staff
CMMI	Capability Maturity Model Integration
COTS	Commercial Off-the-Shelf
CTE	Critical Technology Element
DAB	Defense Acquisition Board
DAPA	Defense Acquisition Performance Assessment
DDR&E	Director of Defense Research & Engineering
DoD	Department of Defense
DoDAF	DoD Architecture Framework
GIG	Global Infrastructure Grid
GOTS	Government Off-the-Shelf
GUI	Graphical User Interface
HDBK	Handbook
ICE	Independent Cost Estimate
IEC	International Electro-technical Commission
IEEE	Institute of Electrical and Electronics Engineers
IOC	Initial Operational Capability
IPA	Independent Program Assessment
IPAT	IPA Team
IRT	Independent Review Team
ISO	International Standards Organization
IT	Information Technology
JWS	Joint War-fighting Space
KDP	Key Decision Point
KIP	Key Interface Profiles
MDA	Milestone Decision Authority
MIL	Military
MMCS	Multimedia Conferencing System

MOSA	Modular Open System Architecture
MPS	Message Passing Service
NCI	Net-Centric Infrastructure
NCO	Net-Centric Operations
NCOW	Net-Centric Operations and Warfare
NCW	Net-Centric Warfare
NR-KPP	Net Ready Key Performance Parameter
NSSAP	National Security Space Acquisition Policy
OMG	Object Management Group
OT&E	Operational Test & Evaluation
OUSD	Office of the Under Secretary of Defense
PSP	Personal Software Process
REX	Remote Execution Protocol
RM	Reference Model
RMA	Rate-Monotonic Analysis
RPC	Remote Procedure Call
SAF/AQ	Secretary of the Air Force/Acquisition
SMC	Space and Missile Systems Center
SME	Subject Matter Expert
SOA	Service Oriented Architecture
SoS	System of Systems
SSAP	Space Systems Acquisition Policy
STD	Standard
TCP/IP	Transfer Control Protocol/Internet Protocol
TRA	Technology Readiness Assessment
TRL	Technology Readiness Level
UML	Unified Modeling Language
USAF	United States Air Force
USC	United States Code
V	Version
WBS	Work Breakdown Structure

References

- Carter 2010** Carter, A. B., OUSD/AT&L, Memorandum for Acquisition Professionals, 14 September 2010
- DAPA 2006** Defense Acquisition Performance Assessment (DAPA) Report, March 2006
- DoD 2008** DoD 5000.02, Instructions on the Operation of the Defense Acquisition System, Signed 8 December 2008
- DoD 2009** DoD Technology Readiness Assessment (TRA) Deskbook, July 2009
- DoDAF 2009** DoD/CIO memorandum, DoD Architecture Framework Version 2.0, Signed 28 May 28 2009
- Eslinger 2006** Eslinger, S., Mission Assurance-driven Processes for Software-intensive Ground Systems, Aerospace Technical Report ATR-2006(8056)-1, September 30, 2006
- Foreman 1997** Foreman, J.T., et al, "C4 Software Technology Reference Guide: A," Handbook, CMU/SEI-97-HB-001, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, January 1997
- GAO 2005** GAO-05-301, Defense Acquisitions Assessments of Selected Major Weapon Programs, Government Accountability Office Report, March 2005
- Hantos 2010** Hantos, P., Software Technology Readiness Assessment - Defense Acquisition Guidance with Space Examples, The Aerospace Corporation Technical Report ATR-2010(8404)-4, September 30, 2010
- Houghton 2009** The American Heritage® Dictionary of the English Language, 4th edition, Houghton Mifflin Co., 2009
- IEEE 2000** IEEE Recommended Practice for Architectural Description of Software-Intensive Systems, IEEE Std 1471-2000, September 21, 2000
- ISO/IEC 1996** Information Technology – Open Distributed Processing – Reference Model, ISO/IEC 10746-1, September 15, 1996
- ISO/IEC 2007** Recommended Practice for Architectural Description of Software-Intensive Systems, ISO/IEC 42010:2007
- MAG 2007** Mission Assurance Guide, The Aerospace Corporation Technical Operating Report, TOR-2007(8546)-6018 Rev. A, July 1, 2007
- Minkiewicz 2007** Minkiewicz, A., *The Cost and Business Impact of SOA – Is it Right for You*, A Price Systems Thought Leadership Article on the Price Systems website
- OUSD 2008** OUSD(AT&L) Memorandum for Service Acquisition Executives on the Implementation of Service Oriented Architecture (SOA) within DOD Acquisition Community, 25 July 2008
- Rocke 2006** Rocke, P., et al, *Net Centric Enterprise Services (NCES) Applicability to Service Oriented Architectures within the DOD Satellite Command Community of Interest (COI)*, Ground System Architectures Workshop (GSAW) 2006
- Rumbaugh 1999** Rumbaugh, J., et al, The Unified Modeling Language Reference Manual, Addison-Wesley 1999
- Schuler 2005** Schuler, M. A., *Joint Warfighting Space and C2 of Deployable Space Forces*, High Frontier – The Journal for Space & Missile Professionals, Vol. 1, No. 4, 2005
- USAF 2009** Software Technology Readiness Recommendations, AFISO21/D&SWS/TD-1-12 Final Report, Revised April 30, 2009

Use of Trademarks, Service Marks and Trade Names

Use of any trademarks in this material is not intended in any way to infringe on the rights of the trademark holder. All trademarks, service marks, and trade names are the property of their respective owners.

The clip art on slide 69 is courtesy of Florida's Educational Clearing House (<http://etc.usf.edu/clipart>)